CHAPTER

13

WORKING WITH THE WINDOWS 2000 REGISTRY

After reading this chapter and completing the exercises, you will be able to:

- Understand the function and structure of the Registry
- Describe the purpose of each of the five Registry keys
- Use the Registry editing tools
- Understand the fault-tolerance mechanisms for the Registry
- Back up and restore the Registry

Windows 2000 is a complex operating system that relies upon a dynamic data structure to maintain its configuration and operational parameters. This structure is the hierarchical database known as the **Registry**, which contains most of the control and functional settings for Windows 2000 core elements, services, and native applications, as well as many Microsoft and third-party add-on software products. In this chapter, you learn about the Registry, its structure, and tools to edit and manage the Registry, as well as several values you may consider altering to improve or configure system operation.

Windows Registry Overview

Windows 2000 uses the Registry to provide a database that stores data about a system's configuration in a hierarchical form. The Registry stores information essential to Windows 2000 itself as well as to native applications, added services, and most add-on software products from Microsoft and third-party vendors. The information stored in the Registry is comparable to information stored in initialization (.ini, .dat, .bat, .sys, and so on) files in Windows 3.x or even Windows 95/98. For native Windows 2000 applications, the Registry database takes the place of .ini files, and stores all configuration information. The Registry is not a text file, such as Win.ini or Config.sys from previous operating systems, but rather a multifaceted branch-like grouping of data. Although most Windows 2000 Professional configuration can be performed using the Control Panel applets and the Administration Tools (and in fact, changes made to system configurations through these tools are applied to the Registry database), some settings can be established or changed only by editing the Registry directly, such as the setting for TCPWindowSize, which determines the number of packets transmitted by a system before the receiver will reply with an acknowledgment of receipt. The Registry can be edited by using one of two Registry editors: Regedit and Regedt32, which are discussed in more detail later in the chapter.

Microsoft warns that editing the Registry directly should only be performed when absolutely necessary. If possible, use the Control Panel applets and Administrative Tools to make system modifications rather than manipulating the values in the Registry. Improper editing of the Registry can cause system malfunction or inoperability.

The Registry was designed for programming ease and speed of interaction for processes. This structure, although a bit daunting, is understandable if broken down into its parts. The Registry is divided into keys and subkeys. Each Registry **key** is similar to a bracketed heading in an .ini file, and is the top-level division in the Registry hierarchy. There are five of these highest-level, or root, keys, which start with HKEY to designate their highest-level status. Each key may contain one or more lower-level keys called **subkeys**. Within each subkey, one or more values can exist. A **value entry** is a named parameter or placeholder for a control setting or configuration data. A value entry can hold a single binary digit, a long string of ASCII characters, or a hexadecimal value. The actual data held by a value entry is known as the **value**. Figure 13-1 shows the structure of the Registry contents in Regedit. The left pane shows five root keys, with subkeys displayed for the HKEY_LOCAL_MACHINE key. The right pane shows the value entries for the SYSTEM\ControlSet002\Control subkey.

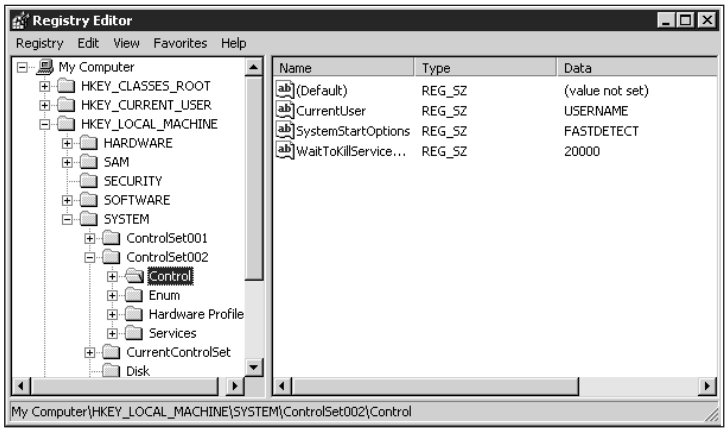

Figure 13-1 View of Registry hierarchy structure

A discrete body of Registry keys, subkeys and values stored in a file is also known as a hive.

Value entries within the Registry are composed of three parts: name, type, and data (value). A Registry value entry's name is typically a multiword phrase, without spaces, with title capitalization, such as AutoAdminLogon in Figure 13-2. The data type of a value entry informs the Registry how to store the value. The **data type** defines whether the data is a text string or a number, and gives the numerical base (radix) of that number. Radix types supported by Windows 2000 are decimal (base 10), hexadecimal (base 16), and binary (base 2). All hexadecimal values are listed with the prefix "0x" to identify them clearly (as in 0xF for 15).

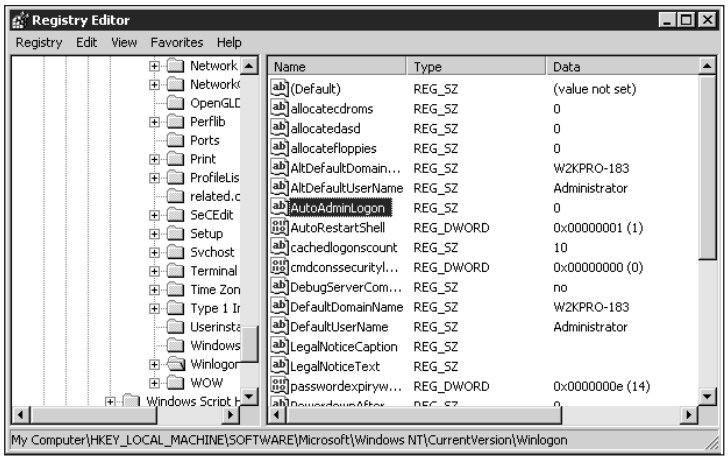

Figure 13-2 AutoAdminLogon value entries

The data types supported by Windows 2000 are:

- **REG_BINARY:** Binary format
- **REG_DWORD:** Binary, hex, or decimal format
- **REG_SZ:** Text-string format
- **REG_MULTI_SZ:** Text-string format that contains multiple human-readable values separated by NULL characters
- **REG_EXPAND_SZ:** Expandable text-string format that contains a variable that is replaced by an application when it is used (%Systemroot%\File.exe)
- **REG_FULL_RESOURCE_DESCRIPTOR:** A resource list for a hardware component or drive, stored as a nested array series (several strings of characters stored in a single value entry)
- **REG_DWORD_LITTLE_ENDIAN:** The same as REG_DWORD, but with a 32-bit number, where the most significant byte is displayed as the leftmost high-order byte
- **REG_DWORD_BIG_ENDIAN:** The same as REG_DWORD, but with a 32-bit number, where the most significant byte is displayed as the rightmost low-order byte
- **REG_LINK:** A symbolic link in the Registry

The cryptic names of these value types do not provide an easy way to remember their meanings. For example, DWORD stands for data representation, and SZ stands for text string.

Once a value entry is created and its data type defined, that data type cannot be changed. To alter the data type of a value, you must delete the value entry then re-create it with a new data type.

Important concepts to keep in mind about the Registry are:

- Keys are the top-level, or root, divisions of the Registry
- Keys contain one or more subkeys
- A subkey can contain one or more subkeys
- A subkey can contain one or more value entries

Also note that the Registry is not a complete collection of configuration settings. Instead, it holds only the exceptions to the defaults. Processes within Windows 2000 will operate with their own internal defaults unless a value in the Registry specifically alters that default behavior. This makes working with the Registry difficult, because most often the control you need is not contained in the Registry because internal defaults are being used. To alter such a setting, you'll need to add a new value entry to the Registry. To accomplish this, you must know the exact syntax, spelling, location, and valid values; otherwise, you will be unable to alter the

default behavior. Keep in mind that not using the exact syntax, spelling, location, and valid values can result in malfunctions, possibly including an inoperable system. So always edit with extreme care. The *Windows 2000 Resource Kit* includes a help file named Regentry.chm, which lists all of the possible Registry entries and valid values. This is an invaluable tool when attempting to modify existing Registry entries as well as when adding new ones.

Each time Windows 2000 boots, the Registry is loaded into memory from files (see "Registry Storage Files" later this chapter) stored on the hard drive. Each time Windows 2000 shuts down, the Registry is written from memory back to the files. While Windows 2000 is operating, the Registry remains in memory. This makes the Registry easy to access and quick to respond to control queries, and it is the reason why changes to the Registry take effect immediately. Only in extreme cases will Windows 2000 require a reboot to enforce changes in the Registry.

Important Registry Structures and Keys

In the following sections, we look at various keys and subkeys of the Registry and explain their functions.

HKEY_LOCAL_MACHINE

The **HKEY_LOCAL_MACHINE** key contains the value entries that control the local computer. These configuration items include information about hardware devices, applications, device drivers, kernel services, and physical settings. This data is used to establish the configuration of the hardware and operating system environment. The content of this key is not dependent on the logged on user or the applications or processes in use; it is dependent only on the physical composition of the hardware and software present on the local computer.

This key has five subkeys (see Figure 13-3): Hardware, SAM, Security, Software, and System, which are detailed in the following sections.

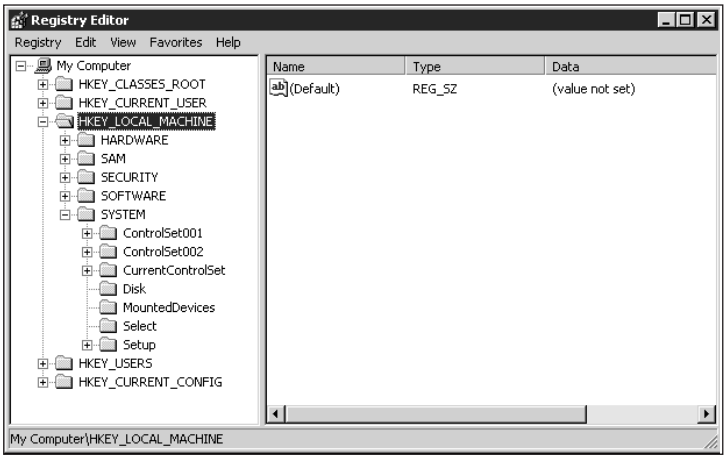

Figure 13-3 HKEY_LOCAL_MACHINE subkeys

HKEY_LOCAL_MACHINE\Hardware

The HKEY_LOCAL_MACHINE\Hardware subkey is the container for data related directly to the installed physical devices of the computer. This subkey stores configuration information, device driver settings, mappings, linkages, relationships between kernel-mode and user-mode hardware calls, and IRQ hooks. This subkey is re-created each time the system boots and is not saved when the system shuts down.

The HKEY_LOCAL_MACHINE\Hardware subkey contains three subkeys: Description, DeviceMap, and ResourceMap. The Description subkey stores data extracted from a device's own firmware or onboard BIOS. The DeviceMap subkey stores information about device driver paths, locations, and filenames. The ResourceMap subkey stores information about the mappings between system resources (I/O ports, I/O memory addresses, interrupts, and direct memory access [DMA] channels) and device drivers. When certain bus types are present in the computer, a fourth subkey named OwnerMap is present. This fourth subkey stores association information about the bus type and device drivers.

The contents of this subkey should not be manipulated. This key contains data read from the state of the physical devices and associated device drivers. Thus there should be no need or reason to alter the data, because it will be a proper reflection of the state of the system. Second, the data is most often in a binary format; thus deciphering the information will be difficult, if not impossible, for most system users. If you wish to view the data contained in this key, you can do so through the System Information tool within the Computer Management utility in Administrative Tools.

HKEY_LOCAL_MACHINE\SAM

The subkey HKEY_LOCAL_MACHINE\SAM is a hive containing data related to security. The **Security Accounts Manager (SAM)** database is stored in this key and is where user accounts and group memberships are defined. The entire security structure of your Windows 2000 system is stored in this key. In most cases, this data is not accessible from a Registry editor.

This is another area of the Registry that you should not attempt to modify. Most of the data contained in this subkey is in binary or encrypted format. You should employ the user manager tools (that is, the Local Users and Groups section of the Computer Management tool) to manipulate the data stored in this subkey. Additionally, to aid in preventing you from editing this subkey, it has a security setting such that only the System (or the System utility) has rights to read and alter the contents of this subkey.

HKEY_LOCAL_MACHINE\Security

The subkey HKEY_LOCAL_MACHINE\Security is the container for the local security policy. The local security policy defines control parameters, such as password policy, user rights, account lockout, audit policy, and general security options for the local machine.

This is yet another area of the Registry that you should not attempt to modify. Most of the data contained in this subkey is in binary format or is encrypted. You should employ the Local Security Policy tool to manipulate the data stored in this subkey (see Chapters 5 and 6). Additionally, to prevent you from editing this subkey, it has a security setting such that only the System has rights to read and alter the contents of this subkey.

HKEY_LOCAL_MACHINE\Software

The subkey HKEY_LOCAL_MACHINE\Software is the container for data about installed software and mapped file extensions. These settings apply to all local users. The \Software\Classes subkey contains the same information as the HKEY_CLASSES_ROOT key; in fact the HKEY_CLASSES_ROOT key is created by copying the data from the \Software\Classes subkey.

HKEY_LOCAL_MACHINE\System

The subkey HKEY_LOCAL_MACHINE\System is the container for the information required to boot Windows 2000. This subkey stores data about startup parameters, loading order for device drivers, service startup credentials (settings and parameters), and basic operating system behavior. This key is essential to the boot process of Windows 2000. It contains subkeys called control sets that include complete information about the boot process for the system. This subkey also contains subkeys that host settings for storage devices (such as MountedDevices) and control set boot status (Select), and possibly subkeys left over from upgrading from Windows NT 4.0 (Disk and Setup). The control set keys are named and numbered, for example, ControlSet001 and ControlSet003. In most cases, there will only be two control sets, and those sets will be numbered 001 and 003. These two sets represent the original (001) system configuration set and a backup (003) of the last functioning system configuration set. Thus, there will always be a functioning configuration to allow the operating system to boot (see Chapter 14 for more information on booting). Each control set has four subkeys:

- *Control:* This control set subkey is the container for data related to controlling system startup, boot parameters, computer name, and necessary subsystems to initiate.
- *Enum:* This control set subkey is the container for data regarding required device drivers and their configuration.
- *Hardware Profiles:* This control set subkey is the container for data specific to the hardware profile currently in use.
- *Services:* This control set subkey is the container for data about drivers, services, file systems, applications, and other required hardware components necessary to load all installed and active services during bootup. This subkey also defines the order in which services are called and the way that one service can call or query other services.

The value entries under the HKEY_LOCAL_MACHINE\System\Select subkey are used to define how the control sets are employed by Windows 2000. The four value entries are:

- *Default:* Defines which control set will be used during the next bootup
- *Current:* Lists the control set that was used to boot the current session
- *LastKnownGood:* Indicates the control set last used to boot and successfully log on a user (see later in this chapter for more details and its use)
- *Failed:* Lists the control set that was replaced by the control set from the LastKnownGood control set because of a failure to boot

The HKEY_LOCAL_MACHINE\System\CurrentControlSet subkey is a redirector to the actual ControlSet### currently in use rather than a truly distinct subkey. This symbolic link is used to simplify the programming interface of applications and device drivers that need information from the active control set. Because of this redirection, when you need to make modifications to the control set, you should use the CurrentControlSet "subkey" to properly direct your changes to the active control set.

HKEY_CLASSES_ROOT

The **HKEY_CLASSES_ROOT** key is the container for information pertaining to application associations based on file extension and COM object data. The contents of this key are copied from the HKEY_LOCAL_MACHINE\Software\Classes subkey. This key is maintained for backward compatibility with legacy applications and device drivers and is not strictly required by Windows 2000 (see Figure 13-4).

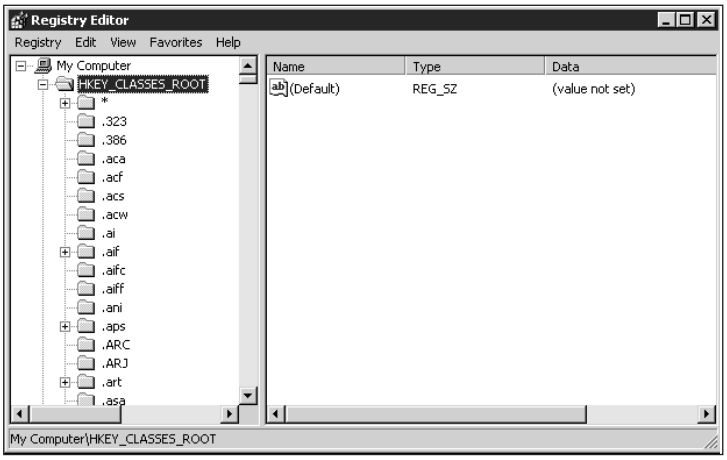

Figure 13-4 HKEY_CLASSES_ROOT

The contents of this key, or the HKEY_LOCAL_MACHINE\Software\Classes subkey, should not be edited directly. Instead, the File Types tab of the Folder Options dialog box should be used. This dialog box is accessed by selecting the Folder Options command from

the Tools menu in Windows Explorer or My Computer or by launching the Folder Options applet from the Control Panel.

HKEY_CURRENT_CONFIG

The **HKEY_CURRENT_CONFIG** key is the container for data pertaining to the hardware profile currently in use. This key is just a symbolic link to the HKEY_LOCAL_MACHINE\System\CurrentControlSet\Hardware Profiles\Current subkey. This key is maintained for backward compatibility with legacy applications and device drivers and is not strictly required by Windows 2000 (see Figure 13-5).

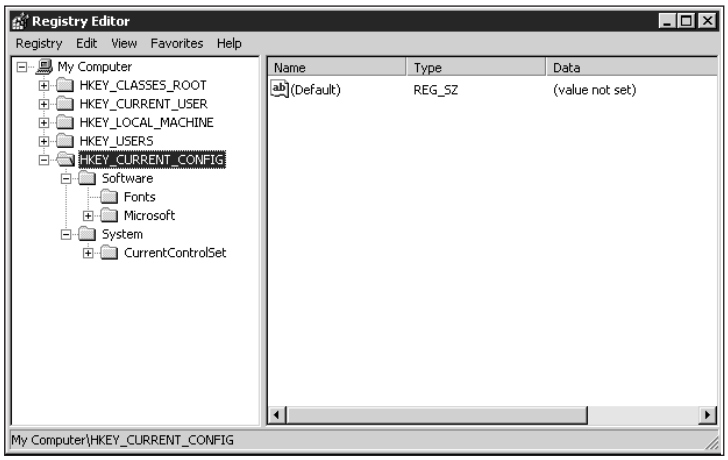

Figure 13-5 HKEY_CURRENT_CONFIG

The contents of this key, and the HKEY_LOCAL_MACHINE\System\CurrentControlSet\Hardware Profiles\Current subkey, should not be edited directly. Instead, the Hardware Profiles interface and Device Manager should be used. The Hardware Profiles interface is accessed by pressing the Hardware Profiles button on the Hardware tab of the System applet from the Control Panel. The Device Manager is accessed by pressing the Device Manager button on the Hardware tab of the System applet from the Control Panel or by selecting the Device Manager node from the Computer Management utility in Administrative Tools.

HKEY_CURRENT_USER

The **HKEY_CURRENT_USER** key is the container for the profile for the currently logged on user. The contents of this key are built each time a user logs on, by copying the appropriate subkey from the HKEY_USERS key. The contents of this key should not be edited directly; instead, you should modify a user's profile through conventional profile management techniques (see Chapter 5 for more information on profile management). (See Figure 13-6.)

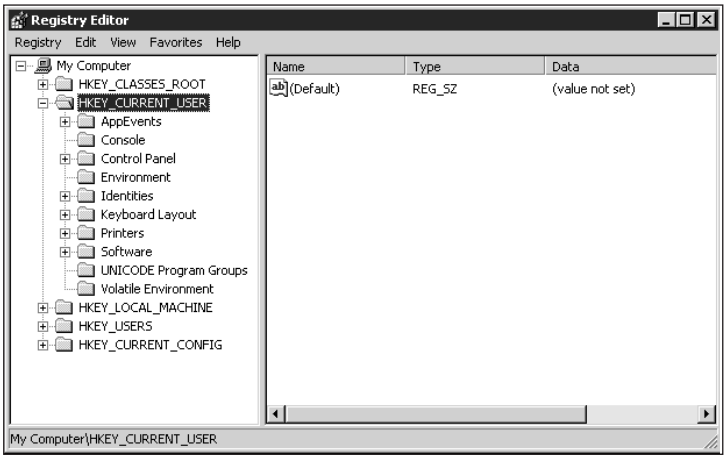

Figure 13-6 HKEY_CURRENT_USER

HKEY_USERS

The **HKEY_USERS** key is the container for profiles for all users who have ever logged on to this system and the default user profile. The contents of this key are built each time the system boots by loading the default file and the locally stored copies of Ntuser.dat or Ntuser.man from user profiles (see Chapter 5). These locally stored copies are found in the \Documents and Settings\<*username*> directory on a Windows 2000 Professional system. To remove a user profile from this key, use the User Profiles tab of the System applet from the Control Panel. To alter the contents of a profile, use conventional profile management techniques (see Chapter 5 for more information on profile management) instead of attempting to edit this key directly (see Figure 13-7).

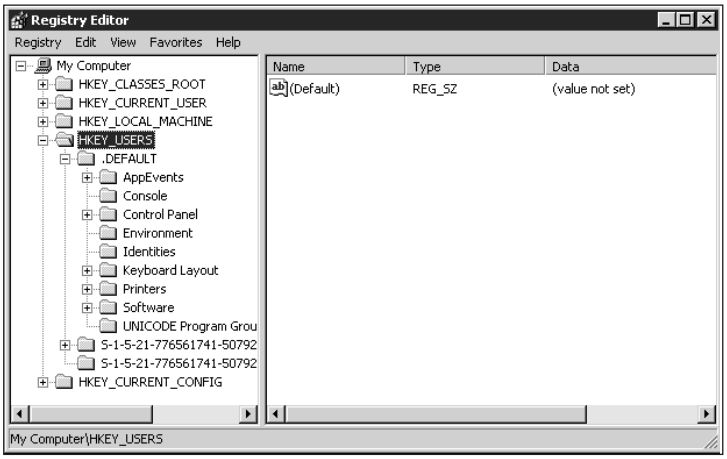

Figure 13-7 HKEY_USERS

THE REGISTRY EDITORS: REGEDIT AND REGEDT32

Because the structure of the Registry is so complex, it requires special tools to operate on directly. In Windows 2000, there are two Registry editors: Regedit and Regedt32. **Regedit** (see Figure 13-8) is a 16-bit application that offers global searching and combines all of the keys into a single display. **Regedt32** (see Figure 13-9) is a 32-bit application that offers access to key and value entry, NTFS-like security, auditing, and ownership, but displays each root key in a separate window (see Hands-on Projects 13-5 and 13-6). Regedt32 also offers a read-only mode so you can explore without the possibility of accidentally altering value entries. Both editors can be used to view keys and values (see Hands-on Project 13-1), perform searches (see Hands-on Project 13-2), add new subkeys and value entries, alter the data in value entries, and import and export keys and subkeys. You'll need to get to know both editors to properly manipulate the Registry.

If you need to locate a specific key or value, use Regedit. If you need to alter the security settings of a key, use Regedt32.

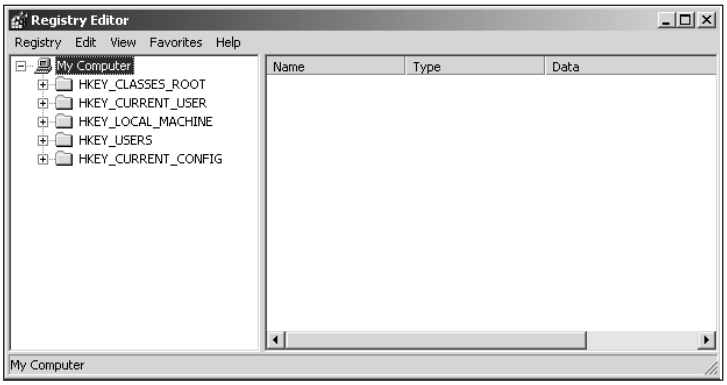

Figure 13-8 Regedit

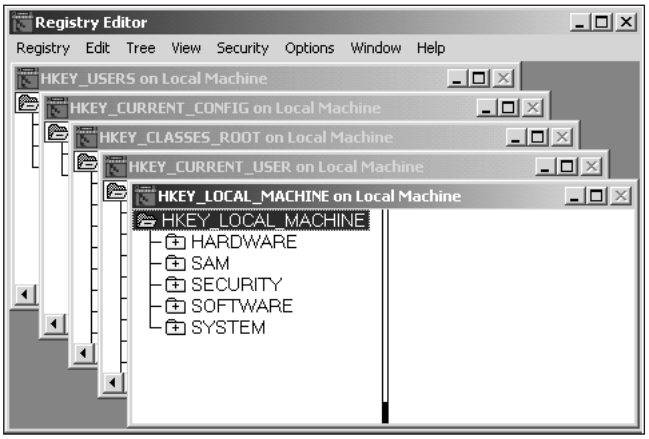

Figure 13-9 Regedt32

As already noted many times in this chapter, editing the Registry directly is a task that should not be undertaken without forethought and planning. It is possible to alter the Registry, whether on purpose or accidentally, in such a way as to render a system completely unrecoverable. If you don't know exactly what you are doing, *don't do it!*

Even when you do think you know exactly what you want to change in the Registry, it is always a good idea to take precautions, such as the following:

- Back up all important data on the computer before editing the Registry.
- Make a distinct backup of all or part of the Registry. Saving each key or subkey individually is recommended (see Hands-on Project 13-3). Store the backup files on local drives, network drives, and floppies or other removable media to ensure access. See the "Backing Up the Registry" section later in this chapter.
- Reboot the machine before editing the Registry.
- Perform only a single Registry modification at a time. Test the results before proceeding.
- Reboot immediately after each change to force full system compliance. This is not strictly necessary, but has often proven a prudent measure.
- Always test changes on a nonproduction system hosting noncritical services before deploying on production systems.
- Use the Regedt32 read-only mode to explore the Registry to ensure that changes are not made accidentally.

Registry Size Limitations

The Registry is stored in active memory so that it is quickly and easily accessible while the operating system is functioning. It resides in the paged pool portion of memory, which means it can be swapped out to disk when not in use. This is unlike the kernel, which resides in a nonpaged pool portion of memory so that it always stays in physical RAM. As your system ages, many changes will accumulate in the Registry, causing the size of the Registry to grow. The initial size of the Registry on a Windows 2000 Professional system is around 10 MB. In order to prevent the Registry from consuming too much memory, Windows 2000 imposes a maximum size for the Registry. This ceiling is set at one-quarter of the current paged pool by default, but can be changed. When the page pool size changes, Windows 2000 will automatically adjust the Registry size ceiling as well. When this value is set to 0, the system automatically creates a maximum Registry size of 33% of the paged pool size. If this value is set to more than 80% of the paged pool size, the system will set the limit to only 80% of the paged pool size.

This value should be changed only when you are prompted to do so by the system. Such a prompt will appear in a message box when the Registry size nears the defined limitation. In most cases, when prompted by the system, increasing the Registry size limitation by 1 or 2 MB is sufficient.

To alter the Registry ceiling, open the Virtual Memory dialog box shown in Figure 13-10 (start from the Control Panel System applet, click the Advanced tab, click the Performance Options button on the Advanced tab, and click Change). Change the value in the number field beside Maximum registry size (MB) within the Registry size area. This value sets only the maximum boundary size for the Registry; it does not allocate paged pool space nor does it guarantee that paged pool space will even be available.

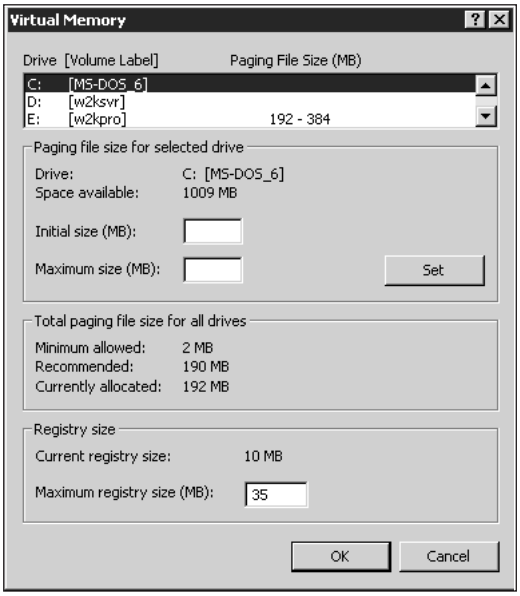

Figure 13-10 Virtual memory dialog box, where the Registry size is defined

Registry Storage Files

The files used to store the Registry are located in the %systemroot%\system32\config and %systemroot%\repair directories of the boot partition (see Figure 13-11). The Registry is not stored in files that match one to one with the top-level keys.

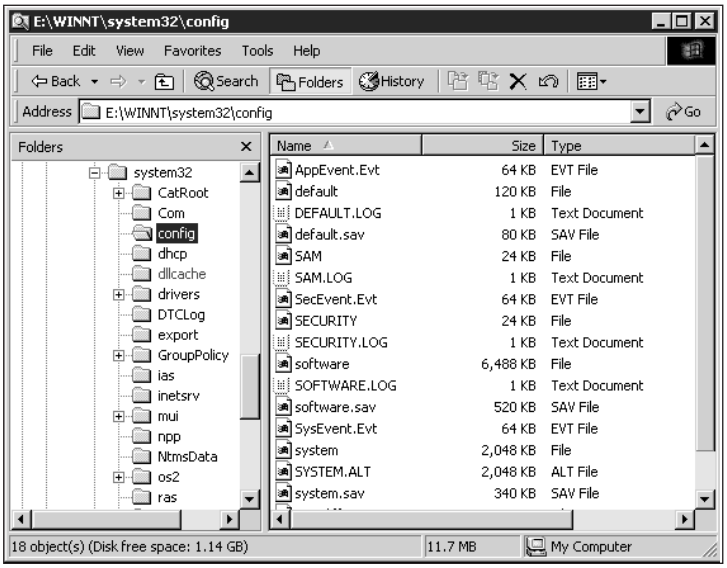

Figure 13-11 The contents of the %systemroot%\system32\config directory

The Registry is stored in various subkey, logging, and backup files, as shown in Table 13-1.

Table 13-1 Registry Storage Files

Registry Key/Subkey	Storage Files
HKEY_LOCAL_MACHINE\SAM	Sam, Sam.log, Sam.sav
HKEY_LOCAL_MACHINE\Security	Security, Security.log, Security.sav
HKEY_LOCAL_MACHINE\Software	Software, Software.log, Software.sav
HKEY_LOCAL_MACHINE\System	System, System.alt, System.log, System.sav
HKEY_USERS\.DEFAULT	Default, Default.log, Default.sav
(Not directly associated with a Registry key)	Userdiff, Userdiff.log
HKEY_CURRENT_USER	Ntuser.dat, Ntuser.dat.log

Note that only four of the HKEY_LOCAL_MACHINE subkeys, the Default subkey of the HKEY_USERS key, and the HKEY_CURRENT_USER key are stored in files. All of the other keys and subkeys are either built "on the fly" at bootup or are copies of a subsection of HKEY_LOCAL_MACHINE.

The HKEY_USERS key is built from the Default file (which represents the default user profile's Ntuser.dat file) and copies of all of the profiles for users who have ever logged on to the computer. These profiles are cached locally in the \Documents and Settings\<*username*> directory. A copy of the Ntuser.dat or Ntuser.man file is copied into the repair directory for the currently logged on user.

Notice that four extensions are used by the Registry storage files. These extensions identify the purpose or function of the file:

- *No extension:* The storage file for the subkey
- *.alt:* The backup file of the subkey. Note that only the HKEY_LOCAL_MACHINE\System subkey has a backup file.
- *.log:* A file containing all changes made to a key. This file is used to verify that all modifications to the Registry are properly applied.
- *.sav:* Copies of keys in their original state as created at the end of the text portion of Windows 2000 installation

Under Windows NT 4.0, the Registry files stored in the \Config directory were used to build the emergency repair disk (ERD). Under Windows 2000, these files are no longer copied onto the ERD when it is created. However, you can create your own custom ERD by manually copying the files in the \Config directory to a formatted floppy. You may find having a complete copy of the Registry quite handy when you need to perform a system repair or restore any portion of the Registry because of corruption or human error.

Registry Fault Tolerance

If the Registry becomes corrupted or destroyed, Windows 2000 cannot function or even boot. Several mechanisms have been established to prevent the Registry from becoming damaged or to automatically repair minor problems. The fault tolerance of the Registry is sustained by its structure, memory residence, and transaction logs. These mechanisms ensure that all changes or operations performed on the Registry either succeed or fail. This prevents any partially applied alterations that would result in an invalid value entry or entries. This the "all or nothing" guarantee is supported no matter what method of alteration is used, including using a Registry editor or an administration tool, or alterations by an application. If the change action is interrupted (by power failure, too little CPU time, hardware failure, etc.), the Registry will remain intact, even if the desired change was not implemented.

As previously mentioned, when an alteration is made to a value entry in the Registry, that change is made to the Registry in active memory. This means that the change affects the system immediately in most cases. The change to the Registry is only made permanent when the key files are written back to the hard drive. This activity occurs during a flush. A **flush** is a copy procedure to update the files on the hard drive with the new settings stored in the memory-resident version of the Registry. A flush occurs at shutdown, when forced by an application, or just after a Registry alteration.

Transaction logs are files where the system records edits, changes, and alterations to the Registry, similar to a list of orders or commands. When a flush occurs, the transaction log is updated to record all changes currently in memory, which will be stored to the Registry storage files. This log is used by the system to automatically verify that the Registry changes are complete once the flush is concluded.

A flush follows the following sequence of steps:

1. All alterations to a key are appended to that key's transaction log file (.log).
2. The key file is marked as being in transition.
3. The key file is updated with the new data from memory.
4. The key file is marked as complete.

If a system failure occurs between the time the key file is marked as in transition and when it is marked complete, the original state of the key is recovered using the data from the transaction log. If the flush finishes uninterrupted, then the system continues to perform normally.

The flush operation is performed on all keys except the System key. This key contains system-critical data and is a major element in a successful bootup of Windows 2000. For this reason, recovery cannot rely upon transaction logs. Instead, Windows 2000 updates the System key with a different method:

1. The system file is marked as being in transition.
2. The system file is brought up to date with the state of the Registry from memory.
3. The system file is marked as being complete.
4. The System.alt file is marked as being in transition.
5. The System.alt file is brought up to date with the state of the Registry from memory.
6. The System.alt file is marked as being complete.

This dual-file process, with the primary and backup copies of the System key file, ensures that, no matter at which stage the update process might be interrupted, a complete and functional copy of the System key file is available. If the failure occurs within the first three steps, then the nonupdated System.alt file will be used to boot. If the failure occurs within the last three steps, then the updated system file will be used to boot. Once booting is complete after a failure, Windows 2000 will perform the update again to ensure that both copies of the system key are exactly the same. However, if the failure occurred during the first three steps, any changes made to the system will have been lost.

Backing Up the Registry

Even though Windows 2000 automatically manages the safety of the Registry via its fault-tolerance mechanisms (.log and .alt files), it is still important for you to take proactive measures to back up the Registry. There are several methods you can employ to create reliable Registry backups:

- Most Windows 2000 backup applications (for example, the built-in Backup tool and third-party products such as Veritas Backup Exec and Stac Replica) include support for full Registry backups. With these products, you can back up the Registry as part of your daily automated backup or as a distinct Registry-only procedure. Backing up the Registry with most of these products consists of selecting a "Back up the Registry" or "System State" (see Figure 13-12) check box when you make file/folder selections before initializing a backup.

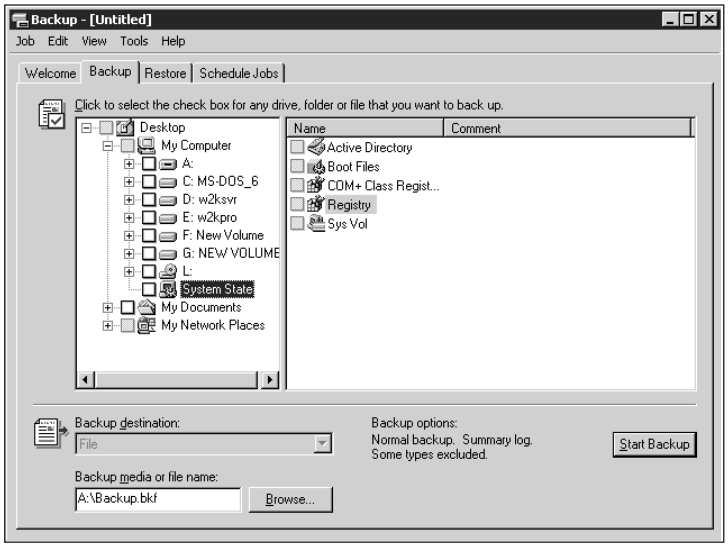

Figure 13-12 Native backup tool showing System State selection

- Regedit or Regedt32 can be used to save all or part of the Registry to distinct files. Both of these tools have an Export or Save command, which is used to save the entire Registry, a single key, or any subportion of a key to a file (try Hands-on Project 13-3).
- Make a copy of the %systemroot%\System32\config and %systemroot%\repair directories manually. Just copy the contents to another location on your local computer, network, or to a floppy disk (if size permits).
- Employ the *Windows 2000 Resource Kit* tools Reg.exe or Regback.exe. Both of these tools offer command-line scripting capabilities. Please explore the *Windows 2000 Resource Kit* for ideas on how to best employ these tools. You can see a syntax

parameter listing for these and most command-line tools by issuing a "/?" parameter after the command from a Command Prompt (that is, *reg /?*, or *reg /? | more* if more than one screen's worth of data is displayed).

No matter what backup method you employ, take the time to make two copies or perform the backup twice. This will provide additional insurance just in case your first backup fails.

Restoring the Registry

Obviously, if you are going to take the time to create backups of the Registry, you need to understand how to restore the Registry. You have several options for restoring the Registry, depending on the method used to make a backup. Windows 2000 itself will attempt to maintain a functional Registry, using its own internal automatic fault-tolerance mechanisms. In cases where the automatic restoration process fails, you can first attempt to restore the Last Known Good Configuration. The **Last Known Good Configuration (LKGC)** is the state of the Registry stored in one of the control sets (see earlier this chapter) when the last successful user logon was performed. If the Registry is damaged in such a way that it will not fully boot or will not allow a user to log on, the LKGC option can restore the system to a previous state.

This boot option is accessed by pressing F8 during the initial bootup of Windows 2000 when the boot menu is displayed. Don't worry, the basic boot menu even prompts you to press F8 if you need an alternate boot method. Pressing F8 reveals a new selection menu similar to the following:

```
OS Loader v5.0

Windows 2000 Advanced Options Menu
Please select an option:

Safe Mode
Safe Mode with Networking
Safe Mode with Command Prompt

Enable Boot logging
Enable VGA Mode
Last Known Good Configuration
Directory Services Restore Mode (Windows 2000 domain
controllers only)
Debugging Mode

Boot Normally
Return to OS Choices Menu

Use [up] and [down] to move the highlight to your choice.
Press Enter to choose.
```

Use the arrow keys to highlight the Last Known Good Configuration selection, then press Enter. Keep in mind that any changes made to the system between the time the LKGC was stored and its use to restore the system will be lost. If the LKGC fails to restore the system to a functioning state, then you only have two options:

1. Use your backup software to restore the Registry files. This is only possible if your backup application offers a DOS-based restore mechanism that can bypass NTFS write restrictions. In other words, the backup software must operate without a functional Windows 2000 environment when launched from a bootable floppy. This type of software lets you restore files to the boot and system partitions (such as the Registry) so you can return to a functional OS. Unfortunately, these applications are few. One such product is Replica from Stac (*www.stac.com*).
2. Reinstall Windows 2000, either fully or as an upgrade. An upgrade may replace the section of the Registry that is causing the problems, allowing you to retain most of your configuration, but this is not guaranteed. A full, new installation of Windows 2000 will return the system to a preconfigured state, requiring you to perform all postinstall changes again.

If you are able to boot into the system, but things are not functioning the way they should or services, drivers, or applications are not loading or operating properly, you may need to restore the Registry in part or whole from backup. Simply use the same tool employed to create the backup to restore the Registry. Keep in mind that with some tools, you can restore portions of the Registry instead of the entire database (see Hands-on Project 13-4).

No matter what method you employ to restore the Registry, it's always a good idea to reboot the system to ensure that the restoration completed successfully and that the system is using only the updated (or more correctly reverted to) settings. It's also a good idea to retain the copies of the old Registry until you are confident that the system is functioning normally, and you've had the opportunity to create new backups. In other words, don't throw away the disks, erase the drives, or format the tapes containing the Registry backup; keep a few generations of Registry backups on hand just in case.

Windows 2000 Resource Kit Registry Tools

The *Windows 2000 Resource Kit* includes several tools that can be used to manipulate the Registry. The *Windows 2000 Resource Kit* is a product release from Microsoft that's separate from the Windows 2000 operating system product. The *Windows 2000 Resource Kit* has additional documentation on Windows 2000, its operations, and its use, as well as a host of useful tools and utilities not included with the shipped software. You can purchase the *Windows 2000 Resource Kit* from Microsoft and most software and book vendors online and off.

Because many of these tools are command-line tools or have significant ancillary materials, we recommend that you peruse the *Windows 2000 Resource Kit* documentation yourself before actually using these tools. Some of the key utilities include:

- *Reg.exe:* Used to perform command line operations on the Registry, ideally suited for batch file operations. Functions include query for value entry data, adding new value entries, changing current values, deleting values or keys, copy keys, back up and restore keys, and load and unload keys. This tool is useful when you need to search for a specific value entry, add new values entries, change an existing value, remove values or keys, create copies of keys, back up or restore keys, and load and unload keys stored in memory.
- *Regdump.exe:* A command-line tool used to dump all or part of the Registry to Stdout. The output of this tool is suitable for the Regini.exe tool. This tool is useful when you need to create scripts based on Registry content by creating a dump of existing settings.
- *Regfind.exe:* A command-line tool used to search for a key, value name, or value data based on keywords. This tool is useful when you need to search for a specific keyword/string in the Registry.
- *Compreg.exe:* A GUI tool used to compare two local or remote Registry keys and highlight all differences. This tool is useful when you need to find the differences between two Registry keys.
- *Regini.exe:* A command-line scripting tool used to add keys into the Registry. This tool is useful when you need to script the amendment of keys to the Registry.
- *Regback.exe:* A command-line tool used to back up keys from the Registry. This tool is useful when you need to create a backup of the Registry via a script.
- *Regrest.exe:* A command-line tool used to restore keys to the Registry. This tool is useful when you need to restore keys to the Registry via a script.
- *Scanreg.exe:* A GUI tool used to search for a key, value name, or value data based on keywords. This tool is useful when you need to search the Registry for a specific keyword or text string.

Chapter Summary

- The Windows 2000 Registry is a complex structure consisting of keys, subkeys, values, and value entries.
- The Registry should be manipulated with extreme caution. Unless absolutely necessary, the Registry should not be edited directly; instead, employ the Control Panel applets and Administration Tools to modify system settings.
- Windows 2000 maintains a functional Registry through several fault-tolerant measures, including transaction logs and backup of key files.

- The Registry is divided into five main keys. The primary and most important key is HKEY_LOCAL_MACHINE, because it hosts data ranging from system startup information to driver settings to the security database.
- Windows 2000 includes two Registry editors, Regedit and Regedt32. The former is useful for global searches, the latter useful for changing security settings on keys and value entries.
- As part of your normal system maintenance and administration, you should create copies of the Registry. Backing up the Registry often is the only way to ensure you have a functional Registry to restore in the event of a failure.

KEY TERMS

data type — The setting on a Registry value entry that defines the data format of the stored information.

flush — The activity of forcing the memory-resident copy of the Registry to be written to files stored on the hard drive. A flush occurs at shutdown, when forced by an application, or just after a Registry alteration.

hive — A discrete body of Registry keys, subkeys, and values stored in a file.

HKEY_CLASSES_ROOT — This Registry key contains the value entries that control the relationships between file extensions (and therefore file format types) and applications. This key also supports the data used in object linking and embedding (OLE), COM object data, and file-class association data. This key actually points to another Registry key named HKEY_LOCAL_MACHINE\Software\Classes, and provides multiple points of access to make itself easily accessible to the operating system itself and to applications that need access to the compatibility information already mentioned.

HKEY_CURRENT_CONFIG — This Registry key contains the value entries that control the currently active hardware profile. The contents of this key are built each time the system is booted. This key is derived from data stored in the HKEY_LOCAL_MACHINE\System\CurrentControlSet\HardwareProfiles\Current subkey. This key exists to provide backward-compatibility with Windows 95/98 applications.

HKEY_CURRENT_USER — This Registry key contains the value entries that define the user environment for the currently logged on user. This key is built each time a user logs on to the system. The data in this key is derived from the HKEY_USERS key and the Ntuser.dat and Ntuser.man files of a user's profile.

HKEY_LOCAL_MACHINE — This Registry key contains the value entries that control the local computer. This includes hardware devices, device drivers, and various operating system components. The data stored in this key is not dependent on a logged on user or the applications or processes in use.

HKEY_USERS — This Registry key contains the value entries that define the user environments for all users who have ever logged on to this computer. As a new user logs on to this system, a new subkey is added for that user, which is either built from the default profile stored in this key or from the roaming user profile associated with the domain user account.

key — A top-level division of the Registry. There are five keys in a Windows 2000 Registry. A key can contain subkeys.

Last Known Good Configuration (LKGC) — The state of the Registry stored in one of the control sets when the last successful user logon was performed. If the Registry is damaged in such a way that it will not fully boot or will not allow a user to log on, the LKGC option can restore the system to a previous state. Keep in mind that any changes made to the system between the time the LKGC was stored and its use to restore the system will be lost.

REG_BINARY — A Registry value entry data type that stores data in binary format.

REG_DWORD — A Registry value entry data type that stores data in binary, hex, or decimal format.

REG_EXPAND_SZ — A Registry value entry data type that stores data in expandable text-string format that contains a variable that is replaced by an application when it is used (for example, %Systemroot%\File.exe).

REG_MULTI_SZ — A Registry value entry data type that stores data in text-string format that contains multiple human-readable values separated by null characters.

REG_SZ — A Registry value entry data type that stores data in text-string format.

Regedit — The 16-bit Registry editor. Regedit offers global searching and combines all of the keys into a single display. It can be used to perform searches, add new subkeys and value entries, alter the data in value entries, and import and export keys and subkeys.

Regedt32 — The 32-bit Registry editor. Regedt32 offers control over key and value entry security, but displays each root key in a separate window. Regedt32 also offers a read-only mode so you can explore the Registry without the possibility of accidentally altering value entries. It can be used to perform searches, add new subkeys and value entries, alter the data in value entries, and import and export keys and subkeys.

Registry — The hierarchical database of system configuration data, which is essential to the health and operation of a Windows 2000 system.

Security Accounts Manager (SAM) — The database of user accounts, group memberships, and security related settings.

subkey — A division of a Registry key, such as HKEY_LOCAL_MACHINE. A subkey can contain other subkeys and value entries.

transaction log — A file created by Windows 2000 to record Registry changes. These files, with a .log extension, are used to verify that changes to the Registry are made successfully.

value — The actual data stored by a value entry.

value entry — A named Registry variable that stores a specific value or data string. A Registry value entry's name is typically a multiword phrase without spaces and with title capitalization.

Review Questions

1. The Registry is the primary mechanism for storing data about Windows 2000. Which of the following are configuration files used by other Microsoft operating systems and may still exist on Windows 2000 for backward-compatibility? (Choose all that apply.)
   a. Win.ini
   b. Autoexec.bat
   c. System.ini
   d. Config.sys
2. The Registry is only used to store configuration data for native Windows 2000 applications, services, and drivers. True or False?
3. Which of the following tools are most highly recommended by Microsoft for editing the Registry?
   a. Control Panel applets
   b. Regedit
   c. Reg.exe
   d. Administrative Tools
4. The Registry is an exhaustive collection of system control parameters. True or False?
5. When editing the Registry, especially when attempting to alter the unseen defaults, which of the following pieces of information are important? (Choose all that apply.)
   a. syntax
   b. spelling
   c. subkey location
   d. valid values
   e. time zone
6. Changes made to the Registry never go into effect until the system is rebooted. True or False?
7. The Windows 2000 Registry has how many default keys?
   a. 2
   b. 4
   c. 5
   d. 6
8. Which of the following can host subkeys or value entries?
   a. data type
   b. key
   c. subkey
   d. value data

9. Each of the highest-level keys of the Registry is stored in a distinct file on the hard drive. True or False?
10. Which Registry key contains the value entries that control the local computer?
    a. HKEY_LOCAL_MACHINE
    b. HKEY_CLASSES_ROOT
    c. HKEY_CURRENT_CONFIG
    d. HKEY_USERS
11. Which Registry key contains the value entries that define the user environment for the currently logged on user?
    a. HKEY_LOCAL_MACHINE
    b. HKEY_CLASSES_ROOT
    c. HKEY_CURRENT_CONFIG
    d. HKEY_CURRENT_USER
12. Which Registry key contains the value entries that control the relationships between file extensions (and therefore file format types) and applications?
    a. HKEY_LOCAL_MACHINE
    b. HKEY_CLASSES_ROOT
    c. HKEY_CURRENT_CONFIG
    d. HKEY_USERS
13. Which Registry key contains the value entries that control the currently active hardware profile?
    a. HKEY_LOCAL_MACHINE
    b. HKEY_CLASSES_ROOT
    c. HKEY_CURRENT_CONFIG
    d. HKEY_CURRENT_USER
14. From which key can you delete subkeys, using the System applet?
    a. HKEY_LOCAL_MACHINE
    b. HKEY_CLASSES_ROOT
    c. HKEY_CURRENT_CONFIG
    d. HKEY_USERS
15. Some Windows 95 applications require a sixth Registry key. Windows 2000 adds the _____ key to maintain backward compatibility, which is actually a redirector rather than an actual key.
16. After you've created a value entry, you can easily change its data type via the Edit dialog box. True or False?

17. The value entry data type that can store binary, hex, or decimal formatted data is _____.
    a. REG_SZ
    b. REG_DWORD
    c. REG_MULTI_SZ
    d. REG_EXPAND_SZ
18. Where are the files used to load the Registry at bootup stored on a Windows 2000 system?
    a. %systemroot%\config
    b. %systemroot%\system32\config
    c. %systemroot%\system\config
    d. %systemroot%\system32\repair
19. Which subkey of HKEY_LOCAL_MACHINE is the only subkey to have a backup file?
    a. SAM
    b. Software
    c. System
    d. Security
20. The process of pushing Registry changes from memory to a hard drive file is known as _____.
21. Which type of file (specified by file extension) is used by Windows 2000 to record the changes to the Registry for verification purposes?
    a. .alt
    b. .sav
    c. .dat
    d. .log
22. Assume that your system is performing an update to the System subkey. While altering the system file, before working on the System.alt file, a system crash occurs. When the system reboots, which of the following will occur?
    a. You'll be prompted whether to use the system or System.alt set of configuration parameters.
    b. The state of the Registry before changes to the System subkey will be restored.
    c. The state of the Registry after changes to the System subkey will be restored.
    d. The system will fail to boot because of a corrupt system subkey.
23. Which subkey usually cannot be edited with a Registry editor?
    a. Hardware
    b. Software
    c. SAM
    d. CurrentControlSet

24. Which ControlSet subkey is the container for data related to controlling system startup, boot parameters, computer name, and necessary subsystems to initiate?
    a. Control
    b. Enum
    c. Hardware Profiles
    d. Services
25. Which subkey of HKEY_LOCAL_MACHINE\System\Select indicates the control set that was last used to boot and successfully log on a user?
    a. Default
    b. Currect
    c. LastKnownGood
    d. Failed

HANDS-ON PROJECTS

Project 13-1

This hands-on project is useful when you want to view the current value of a value entry or to determine if a value entry is even present in the Registry.

To view Registry value entries with Regedit:

1. Click the **Start** menu, then click **Run**.
2. Type **regedit**, then click **OK**. The Registry Editor opens.
3. Double-click **HKEY_LOCAL_MACHINE**.
4. Locate and double-click **SOFTWARE** under HKEY_LOCAL_MACHINE.
5. Locate and double-click **Microsoft** under SOFTWARE.
6. Locate and double-click **Windows NT** under Microsoft.
7. Locate and double-click **CurrentVersion** under Windows NT.
8. Locate and select **Winlogon** under CurrentVersion.
9. In the right pane, locate and select **DefaultUserName**.
10. From the Edit menu, select **Modify**.
11. Notice that the value of this value entry is the name of the user account you are currently using.
12. Click **Cancel**.
13. In the left pane, scroll up until you see HKEY_LOCAL_MACHINE.
14. Double-click **HKEY_LOCAL_MACHINE**. Leave the system as is for the next hands-on project.

Project 13-2

To search for a value entry with Regedit:

This hands-on project requires that Hands-on Project 13-1 be completed. In this project, you use Regedit to locate a key or value without knowing its path within the Registry. This hands-on project begins at the system status point where the previous hands-on project ended.

1. From the Edit menu, select **Find**.
2. In the Find what field, type **DefaultUserName**.
3. Click **Find Next**. After a few seconds of searching, Regedit will locate the first key, value, or data containing that string.
4. Notice that the first found match is AltDefaultUserName.
5. From the Edit menu, select **Find Next**. The item found now is the actual DefaultUserName value entry that you viewed in Hands-on Project 13-1.
6. In the left pane, scroll up until you see HKEY_LOCAL_MACHINE.
7. Double-click **HKEY_LOCAL_MACHINE**. Leave the system as is for the next hands-on project.

Project 13-3

The ability to make backups of the Registry offers you an additional level of support in the event of a system problem or a human error in regard to the Registry. Plus, backing up the Registry is always a good action to take before beginning any modifications to the Registry, either through the manual tools of Regedit and Regedt32 or through any of the Control Panel or Administrative Tools utilities.

To back up a Registry key:

This hands-on project begins at the system status point where the previous hands-on project ended.

1. Make sure that the HKEY_USERS key is selected.
2. From the Registry menu, select **Export Registry File**.
3. Select a destination folder (such as c:\Temp) of your choice.
4. Provide a filename, such as **HUsave.reg**.
5. Make sure the **Selected branch** radio button at the bottom of the Export Registry File dialog box is selected and that HKEY_USERS is listed in the text field.
6. Click **Save**.

7. The Regedit tool will create a backup file of the selected key. Leave the system as is for the next hands-on project.

This procedure can be used on larger or smaller portions of the Registry simply by selecting different keys or subkeys. In other words, this process can be used to back up the entire Registry or just a small subset of subkeys.

Project 13-4

If you have made any change to the system or Registry since Hands-on Project 13-3 was performed, you may not want to perform this project because it will discard those changes by restoring the state of the Registry from the saved file.

To restore a Registry key:

This hands-on project requires that Hands-on Project 13-3 be completed. It begins at the system status point where Hands-on Project 13-3 ended.

1. From the Registry menu, select **Import Registry File**.
2. Locate and select your **HUsave.reg** file.
3. Click **Open**.
4. After a few moments of importing, a message stating whether the import succeeded is displayed; click **OK**.
5. From the Registry menu, select **Exit**.

This procedure can be used on larger or smaller portions of the Registry simply by selecting different keys or subkeys. In other words, this process can be used to restore the entire Registry or just a small subset of subkeys. However, it does require that the same or the same plus more data be backed up for the material to be restored.

Project 13-5

To use Regedt32:

1. Click the **Start** button, select **Run** to open the Run command dialog box.
2. Type **regedt32**, then click **OK**. The Registry Editor opens.
3. Notice that each key is displayed in a separate window within the Registry editor.
4. Select **HKEY_LOCAL_MACHINE on Local Machine** from the Window menu.
5. From the View menu, select **Find Key**.
6. Type **WinLogon**.

7. Click **Find Next**. Notice that you are back in the same subkey as was viewed in Hands-on Project 13-1.
8. Click **Cancel** in the Find dialog box to close it.
9. From the Options menu, select **Read Only Mode**. Leave the system as is for the next hands-on project.

Project 13-6

To view security with Regedt32:

This hands-on project requires that Hands-on Project 13-5 be completed.

1. This hands-on project begins at the system status point where the previous hands-on project ended.
2. From the Window menu, select **HKEY_USERS**.
3. From the Security menu, select **Permissions**.
4. A notice may appear stating that you can only view permissions for this key; click **OK**.
5. Notice the Permissions dialog box for the Registry is identical to that used elsewhere in Windows 2000.
6. Click **Cancel**.
7. From the Registry menu, select **Exit**.

CASE PROJECTS

1. Describe the actions that you can perform manually or that are performed automatically that provide protection or fault-tolerance mechanisms for the Windows 2000 Registry.
2. You have been asked to perform several Registry modifications to fine-tune an application. You'll be following detailed instructions from the vendor. What steps can you take to ensure that even if the vendor's instructions fail, you'll be able to return to a functioning Windows 2000 system?